Hybrid Dynamic Surrogate Modelling for a Once-Through Steam Generator

Sindre Stenen Blakseth,[a,b] Leif Erik Andersson,[a] Rubén Mocholí Montañés,[a] Marit Jagtøyen Mazzetti,[a]

[a]*Department of Gas Technology, SINTEF Energy Research, Trondheim, Norway*
[b]*Department of Mathematical Sciences, Norwegian University of Science and Technology, Trondheim, Norway*

Abstract

Four surrogate modelling techniques are compared in the context of modelling once-through steam generators (OTSGs) for offshore combined cycle gas turbines (GTCCs): Linear and polynomial regression, Gaussian process regression and neural networks for regression. Both fully data-driven models and hybrid models based on residual modelling are explored. We find that speed-ups on the order of 10k are achievable while keeping root mean squared error at less than 1%. Our work demonstrates the feasibility of developing OTSG surrogate models suitable for real-time operational optimization in a digital twin context. This may accelerate the adoption of GTCCs in offshore industry and potentially contribute towards a 25% reduction in emissions from oil & gas platforms.

Keywords: Digital Twin, Surrogate Modelling, Residual Modelling, Gaussian Process Regression, Neural Networks

1. Introduction

Installation of combined cycle gas turbines (GTCCs) offshore can reduce emissions from oil & gas platforms by up to 25% [Mazzetti et al. (2014)]. A digital twin (DT) framework for GTCCs may accelerate the adoption of GTCCs by increasing their reliability and performance. To this end, accurate and trustworthy yet computationally efficient dynamic models for crucial GTCC components are needed. The once-through steam generator (OTSG) has been identified as a particularly important component because its response to transients in the gas turbine load largely governs the GTCC's overall behavior. SINTEF Energy Research has previously developed highly accurate Modelica models for OTSGs [Montañés et al. (2021)]. In this work, we explore four data-driven techniques for creating high-speed, DT-suitable surrogate models based on the Modelica model: Linear regression (LReg), polynomial regression (PReg), Gaussian process regression (GPR), and neural networks (NNs). We use these techniques to develop both purely data-driven surrogate models and hybrid models utilizing the residual modelling technique. The goal of this work is to evaluate the feasibility of developing DT-suitable OTSG surrogate models, e.g., for real-time optimization and process control based on model predictive control schemes. The work is intended to contribute towards accelerating the adoption of GTCCs, thereby reducing emissions from offshore energy production. Secondarily, it is of general interest to analyze the relative performance of the different techniques studied.

2. Once-Through Steam Generators

2.1 Background and Motivation

A waste heat recovery steam generator (HRSG) is a heat exchanger boiler system that recovers waste heat from a given heat source by producing steam from feedwater that is circulated through the heat exchanger. In the context of GTCCs, the heat source is the gas

turbine exhaust, and the steam produced by the HRSG is passed through a steam turbine to generate electricity. Consequently, in comparison to standalone gas turbines, GTCCs offer significantly increased efficiency, and thus a corresponding reduction in CO_2 emissions for fixed power production.

In this work, we consider once-through steam generators (OTSGs), which are a particular type of HRSGs common in industrial applications. OTSG configurations are the preferred option in volume- and weight-constrained energy system environments, such as offshore oil and gas installations or floating production, storage and offloading systems. High-fidelity, physics-based, dynamic OTSG models facilitate simulation-based studies to better understand the inherent dynamics of the OTSG system, and to conduct control loop and control structure design studies. Traditionally, operation of OTSG systems has predominantly taken place under steady-state conditions. Therefore, it has been sufficient to consider transient conditions in the offline design phase. Faster models suitable for real-time operational optimization have consequently not received much attention. However, as intermittent renewable energy sources enter the offshore energy mix, computationally efficient models applicable to transient conditions are becoming increasingly important. This motivates the present study of surrogate models for OTSGs, i.e., low-fidelity OTSG models that are designed to capture the main characteristics of OTSG transient behavior while offering significant computational speed-up in comparison to traditional high-fidelity models.

2.2 High-Fidelity OTSG Modelling

Development of surrogate models generally requires data from which the main characteristics of the considered system can be extracted. In the present work, this data will stem from a previously published high-fidelity (hi-fi), dynamic OTSG model for offshore combined cycle applications [Montañés et al. (2021)]. The model is developed in the Modelica language and utilizes dynamic energy and mass balances to produce a set of differential algebraic equations describing the OTSG's transient behavior.

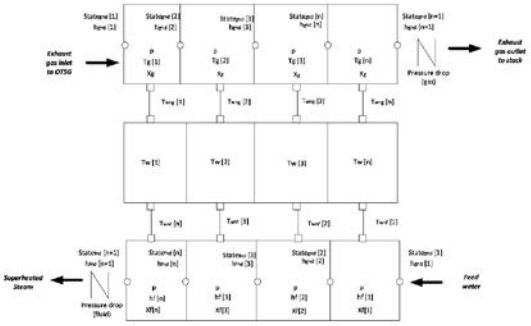

Figure 1: Discretization and main states of high-fidelity OTSG model [Montañés et al. (2021)].

Figure 1 (adapted from [Montañés et al. (2021)]) shows the OTSG model with discretization and main states. The model is based on 1-D lumped pressure flow pipes (for gas and water/steam sides) separated from each other by a wall model (representing the metal wall for heat transfer in a bundle of tubes). A total of *n* volumes and *n+1* nodes are used to discretize the system in the direction of the flow. Overall, the model includes physical phenomena related to heat transfer, hydraulics, wall model thermal inertia, dynamic energy and mass balances on both the gas side and the water side [Montañés et al. (2021)]. We consider the OTSG operated under valve-throttling pressure control

mode, i.e., with fixed pressure set point of the produced superheated steam [Nord and Montañés (2018), Zotică et al. (2022)]. Important boundary conditions for the model are gas turbine load (exhaust gas inlet mass flow and inlet specific enthalpy) and feed water mass flow rate (which is commonly a manipulated variable for OTSG operation).

3. Surrogate Modelling Techniques

In this section, we briefly describe the four surrogate modelling techniques considered in this work: linear and polynomial regression (Section 3.1), neural networks (Section 3.2) and Gaussian Process Regression (Section 3.3). We also describe how we generate the data needed to develop OTSG surrogate models using these techniques (Section 3.4) and how we evaluate the resulting models (Section 3.5). Common for all the surrogate models considered here is that their task is to predict the outlet temperature of the flue gas, as well as the outlet temperature and mass flow of the steam.

3.1 Linear and Polynomial Regression

Polynomial regression (PReg) amounts to approximating some target function $f(\cdot)$ using a polynomial constructed using N_i input data variables I_i, where $i = 1, \dots, N_i$. Mathematically, this can be expressed as finding coefficients $\alpha_{i,n}$ such that

$$\hat{f}_{\text{reg}} \coloneqq a_0 + \sum_i \alpha_{i,1} I_i + \alpha_{i,2} I_i^2 + \cdots + \alpha_{i,N} I_i^d$$

is as close to the true target *f* as possible. We say that *d* is the degree of the interpolating polynomial. Linear regression (LReg) is simply the special case of $d = 1$. We used the Python package lmfit [Newville et al. (2014)] to implement our regression models.

3.2 Neural Networks for Regression

Owing to their universal function approximation properties and acclaimed empirical successes, neural networks (NNs) are well suited for surrogate modelling. Here, we consider fully connected feed-forward NNs (cf. [Nielsen (2015)] for an introduction). Such a network with n_l so-called *layers* can be expressed as

$$\hat{f}_{\text{NN}} \coloneqq \varphi_{n_l}\big(\dots(\varphi_2(\varphi_1(I w_2 + b_1) w_2 + b_2)\dots) w_{n_l} + b_{n_l}\big),$$

where *I* is the NN's input vector, the matrices $w_1, \dots, w_{n_l}$ (weights) and vectors $b_1, \dots, b_{n_l}$ (biases) are tunable parameters, and $\varphi_1, \dots, \varphi_{n_l}$ are non-linear functions known as activation functions. Typically, stochastic gradient descent is used to tune the weights and biases such as to minimize the difference between $\hat{f}_{\text{NN}}$ and the target function *f*. We have implemented our NN models using the Python package pytorch [Paszke et al. (2019]. We use the LeakyReLU activation function with slope parameter 0.01, pytorch's MSELoss cost function, the Adam optimizer, and a learning rate of 1e-5.

3.3 Gaussian Process Regression

Gaussian Processes (GPs) are non-parametric, probabilistic kernel methods [Rasmussen and Williams (2006)] that aim to identify an unknown function $f: \mathcal{R}^{n_u} \rightarrow \mathcal{R}$ from data. It is assumed that the noisy observation of $f(\cdot)$ are given by

$$y = f(\boldsymbol{u}) + v,$$

where the noise v is Gaussian with zeros mean and variance σ_v^2, and $\boldsymbol{u}$ is the input, which is assumed to follow a multivariate Gaussian distribution. Smoothness properties of the underlying function f are enforced by the choice of mean and covariance function without relying on parametric assumptions [Snelson and Ghahramani (2006)]. A zero mean function and the automatic relevance (ARD) squared-exponential (SE) covariance

function are chosen. GP depends on hyperparameters, which are usually unknown and need to be inferred from data. The marginal likelihood is used to estimate hyperparameters. The predictive distribution is the marginal of the normalized joint prior times the likelihood. The integral can be evaluated in closed form. The GP regression model was implemented in Python and the maximisation of the log marginal likelihood was solved with help of the SciPy package [Virtanen et al. (2020)].

3.4 Data Generation

Due to the unavailability of suitable operational data from a real GTCC, the high-fidelity OTSG model described in Section 2.2 will be used to create the reference data needed for developing and evaluating our surrogate models. As explained in Section 2.2, the gas turbine load is an essential boundary condition for the model and must therefore be prescribed. To cover a wide range of operating conditions, a randomized time series of one million data points at 1-second intervals, was generated. Every 100 seconds, a load change is occurring with 50% probability. The type of change (step or ramp), the new set point value and the duration of the transition (if it is a ramp) are randomized with the latter two drawn from uniform distributions on [40%, 100%] and [1s, 300s], respectively.

Using the generated gas turbine load sequence, the hi-fi model is used to generate two time series for each of the three outlet variables we aim to predict with our surrogate models. One time series is created using a coarse discretization in the hi-fi model (we call this the low resolution (LR) data), and the other is created using a fine discretization (denoted high resolution (HR) data). The HR data is used as ground truth, both during the model tuning process and the final evaluation. The LR data serves two purposes. Firstly, it can be used as additional input for the surrogate models. Secondly, it will allow us to explore the potential benefit of residual modelling. The concept of residual modelling is to model the residual $\varepsilon = HR - LR$ instead of the HR data directly [Willard et al. (2022)]. Predictions are then constructed as $\widehat{HR} = LR + \hat{\varepsilon}$, where $\hat{\varepsilon}$ is the model's approximation of ε. Both the HR and LR data are split into three subsets. The first 980k data points are included in the *training set* (used for tuning model parameters), the next 10k data points go in the *validation set* (used for tuning certain hyperparameters), and the final 10k data points constitute the *test set* (used for evaluating the models).

3.5 Model Evaluation

All our surrogate models depend on one or more so-called hyperparameters, i.e., parameters that define a model's structure but are not used directly to compute predictions. Examples include choice of input variables, polynomial degree (PReg), number of layers and neurons per layer (NN) and the number of data points to use for tuning (GPR). To facilitate a fair comparison between the different kinds of models, we conducted a grid search to identify good hyperparameter choices for all models. For each model instance (corresponding to a particular choice of hyperparameters), the normalized root mean squared error (NRMSE) of the models' predictions was computed with respect to the HR test data for each of the three target variables. The sum of the three NRMSEs was taken to represent the overall predictive accuracy of any particular model. For models using old HR data as input, it is important to consider that, when predicting more than one time step into the future, HR data from the previous time step will not be available. Then, the surrogate model's prediction from the previous time step must be used in its place. This may lead to divergent behavior, as is observed for some of our models (cf. Table 1).

4. Results

4.1 Predictive Accuracy

For each modelling technique and model input choice, Table 1 lists the lowest total NRMSE error observed in our grid search for both normal and residual models using the specified technique and input choice. The available input choices are 1) only inlet data (in), 2) inlet data and LR outlet data (in+LR), 3) same as 2) but also including old HR data (in+LR+old), and 4) same as 2), but also including the residual HR-LR (in+LR+res).

We observe that the most accurate model is a GPR residual model using in+LR+old input. However, several other models achieve comparable performance, and the performance different between the best PReg, NN and GPR models is generally insignificant from a practical perspective. Even the best LReg models are found to perform quite well, with a NRSME significantly smaller than 1% achievable for all three output variables. Moreover, while the use of LR data appears to be generally beneficial, we observe that only inlet data is sufficient to obtain NRSME values well below 1% for any given output variable. (Note that, in the table, we sum the NRMSE of each predicted variable.) Figure 2 shows the predictions of steam outlet mass flow made by the best model for each modelling technique (bold in Table 1). Based on Figure 2, it is difficult to identify any predominant failure mode for any of the different models.

	in		in+LR		in+LR+old		in+LR+res	
	Normal	Residual	Normal	Residual	Normal	Residual	Normal	Residual
LReg	0.0555	0.0154	0.0089	0.0089	**0.0088**	0.0514	19.4011	19.3999
PReg	0.0555	0.01525	0.0056	0.0056	***0.0031***	NAN	NAN	NAN
NN	0.0562	0.0107	0.0066	0.0061	0.0039	**0.0032**	0.0406	0.0107
GPR	*0.0369*	*0.0099*	*0.0045*	*0.0043*	0.0032	***0.0031***	*0.0036*	*0.0031*

Table 1: Normalized RMSE on the test set, summed over the three output variables, for given model type and input selection. Bold and italics are used for lowest error values in columns and rows, respectively. NAN indicates that the model diverged.

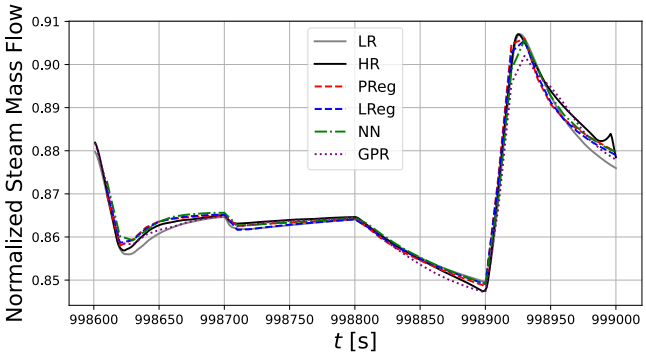

Figure 2: Predictions of steam outlet mass flow by best model within each category, shown along with the corresponding LR and HR data for a short interval within the test set.

4.2 Computational Efficiency

We use the number of CPU seconds (wall time) per simulated second of operations (CPUs/s), as measured on a standard, mid-range laptop, to quantify the computational efficiency of the various models. Using respectively the coarse and the fine spatial discretization, we measure 0.00289 CPUs/s and 0.1272 CPUs/s for the hi-fi-model. Thus, the reduced resolution yields a speed-up of roughly 44. In comparison, our linear regression model with only inlet data as input uses around 8e-6 CPUs/s, which

corresponds to a speed-up of roughly 15k. The PReg model is roughly as fast as the LReg model, while the NN and GPR models are roughly 1–1.5 orders of magnitude slower than the LReg model. Consequently, for residual models and models using LR data as input, the LR hi-fi component always dominates the computational expense.

5. Discussion

An interesting takeaway from our numerical experiment is that blindly relying on more advanced techniques like neural networks and GPR to be better than simple techniques like regression is not advised. The relation between a model's theoretical representation capacity and its empirical predictive accuracy is not necessarily linear. Indeed, our results show that the relation is not even strictly increasing in general, as 1) our best PReg model outperforms our best NN model, and 2) increasing the number of neurons in the NN (and thereby its representation power) did not improve its accuracy on the *test* set.

Our results also show that the value of hybrid modelling depends on the quality of the physics-based model component, and how that component is integrated into the fully hybrid model. In our case, using LR predictions as input was generally observed to be useful, while residual modelling yielded mixed results. This illustrates the obvious, but easily overlooked fact that residual modelling is only beneficial if the relation between the input data and the residual is simpler than that between the input data and the data to be predicted. This criterion is not always met in practice, as evidenced by our results.

Finally, we conclude that it is feasible to construct OTSG surrogate models suitable for use in real-time optimization procedures within a digital twin framework. This motivates further work, which could include exploration of more advanced neural network-based techniques, such as Long Short-Term Memory networks. Additionally, we believe that looking into the robustness of the methods with respect to noisy data would be valuable from a practical perspective.

Acknowledgementss

This work was supported by DIGITAL TWIN, RCN project no. 318899.

References

Mazzetti, M. J., Nekså, P., Walnum, H. T., & Hemmingsen, A. K. T. (2014). Energy-efficiency technologies for reduction of offshore CO2 emissions. Oil and gas facilities, 3(01), 89–96.

Montañés, R. M., Skaugen, G., Hagen, B., & Rohde, D. (2021). Compact Steam Bottoming Cycles: Minimum Weight Design Optimization and Transient Response of Once-Through Steam Generators. Frontiers in Energy Research, 9, 687248.

Newville, M., Stensitzki, T., Allen, D. B., & Ingargiola, A. (2014). LMFIT: Non-Linear Least-Square Minimization and Curve-Fitting for Python (0.8.0). doi: 10.5281/zenodo.598352

Nielsen, M.A. (2015). Neural Networks and Deep Learning. Determination Press.

Nord, L. O., & Montañés, R. M. (2018). Compact steam bottoming cycles: Model validation with plant data and evaluation of control strategies for fast load changes. Applied Thermal Engineering, 142, 334–345.

Paszke, A. et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Advances in Neural Information Processing Systems, 32, 8024–8035.

Snelson, E. & Ghahramani, Z. (2005). Sparse Gaussian processes using pseudo-inputs. Advances in neural information processing systems, 18.

Virtanen, P. et al. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17(3), 261–272

Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2022). Integrating scientific knowledge with machine learning for engineering and environmental systems. ACM Computing Surveys, 55(4), 1–37.

Williams, C. K., & Rasmussen, C. E. (2006). Gaussian processes for machine learning (Vol. 2, No. 3, p. 4). Cambridge, MA: MIT press.

Zotică, C., Montañés, R. M., Reyes-Lúa, A., & Skogestad, S. (2022). Control of steam bottoming cycles using nonlinear input and output transformations for feedforward disturbance rejection. IFAC-PapersOnLine, 55(7), 969–974.